Name: *Student id:* *Section:* *Serial#:*

QUESTION ONE: Convert the following C++ code into equivalent assembly code. {8 pts}

```
int AA[12]={2,-1,5,-6,9,-7,-8,0,4,-3,-2,9}
cout<< "The mystery numbers are: ";
for(j=0; j<12; j++)
  {
    if(AA[j] > 0)
      cout<< AA[j]<< " ";
  }
cout << endl;
```

```
.data
AA[X4]    sword    2,-1,5,-6,9,-7,-8,0,4,-3,-2,9
Str1      byte     "The mystery numbers are: ",0

.code
clrscr
more      edx, offset[Str1]
call      WriteString
mov       eax, (j)=??
mov       eax, 0
cmp       eax, 12
jGE       Next
cmp       AA[eax], 0
jLE       ? (Next)
mov       eax, (j)??
call      WriteInt
Next??
? call crlf
```

```
include irvine32.inc
.data
m1 bybyte "
AA sdword 2,-1,5,-6...
.code
Mov ecx, 12
Mov esi, 0
L1: Mov eax, AA[esi]
CMP eax, 0
jG  L2
INC esi
jmp L1
L2: Call WriteInt.
INC esi
jmp L1
Loop L2
Loop L1
Call Crlf
exit
main endp
end main
```

Name: Student id: Section: Serial#:

QUESTION TWO: Write a sequence of assembly instructions to perform each of the following tasks:

1) Divide the signed values 47890CH / C000H {2 pt}

```
mov   eax, 47890C H
mov   ebx, C000 H
IDiv  ebx
```

2) Set the odd-numbered bits in EBX register to 1. Leave other bits in EBX unchanged {2 pt}

```
pushfd
popfd
eax
ecx
```

OR

```
mov   ebx, 1
popfd
pushfd
ecx
eax
```

3) Inverse the odd-numbered bits in ECX register. Leave other bits in ECX unchanged {2 pt}

```
XOR   ecx,
```

```
XOR   ecx,
```

4) Convert the signed value in AX into a double word in DX:AX {2 pt}

```
mov   ax, val1
CWD
```

```
mov   eax, val1
```

5) Set all bits in the flags register to 1. {2 pt}

```
eax
```

```
push  0FFFFFFFF H
popfd
```

6) Move the unsigned value in ax register into ebx register. {2 pt}

```
mov   ax, value
mov   ebx, value
```

```
movzx  edx, ax
```

7) Store in EBX register the product of multiplying CL register by 1024. CL register may contain any unsigned value. (MUL amd IMUL instructions are not allowed to be used). {2 pt}

```
mov   ebx, 0
SHL   cl, 10
ax
```

```
movzx  eax, cl
SHL    ebx, 10
```

8) Swap the contents of 2 predefined memory words (M3, M7) without using MOV instructions. {2 pts}

```
pushfd
pushfd
eax, M3
eax, M7
```

```
popfd
pushfd
eax, M7
eax, M7
```

```
push  M3
push  M7
pop   M3
pop   M7
```

Name: *Student id:* *Section:* *Serial#:*

QUESTION THREE: {12 pts}

a)
```
MOV  AX, 3AE5H
TEST AX, 0FCD7H
OR   AX, AX
```
AX = FC7A H 3AE5H

b)
```
MOV  AX, 4C9AH
MOV  BX, 768BH
SHRD BX, AX, 8
```
BX = B85 H 9A76

c)
```
MOV  AX, 3A90H
MOV  CX, 4CA0H
IMUL CL
```
AX = 4600 H

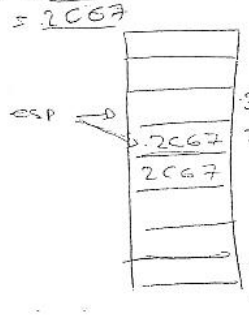
d)
```
MOV AX, 300BH
MOV BX, 2640H
DIV BL
```
AX = 0BC0 H

e) What will be in registers AX and ESP after executing the following instruction sequence?

```
        MOV   ESP, 3000H
        MOV   EAX, -1
        MOV   ECX, 2C67H
        PUSH  ECX
        CMP   AX,CX
        JL    L9
        PUSH  EAX
        AND   AX, 3A4CH
L9:     XOR   AX, CX
```
AX = C3 3A1C H ESP = 2FFC H

f) If "JL L9" is replaced by "JB L9", what will be in registers AX, ESP after executing the above code:

AX = 3A1C H ESP = 2FF8 H



Name: *Student id:* *Section:* *Serial#:*

QUESTION FOUR: {8 pts }

Write a complete assembly program that:

- Prompts the user to enter from the keyboard an integer value in the range 1 - 20 .
- Reads the value and validates it against the above given range limits. If an invalid value is entered, loop until a valid value is entered.
- Calls a procedure **TRI8** that displays a triangle of stars "*" as shown in the example below
- The procedure accepts as a parameter the entered value in eax register.

For example, if you entered 5 from the keyboard, the output should be the following shape of stars:

```
*
**
***
****
*****
```